RECONCILING COMBINATIONS OF TRANSACTIONS

INVENTOR:

DAVID R. LARSEN

PREPARED BY:

AMIR H. RAUBVOGEL
REG. NO. 37,070
FENWICK & WEST LLP
TWO PALO ALTO SQUARE
PALO ALTO, CA 94306
(650) 858-7276

RECONCILING COMBINATIONS OF TRANSACTIONS

Inventor:

David R. Larsen

BACKGROUND OF THE INVENTION

1. Field of the Invention

The present invention relates generally to financial and investment software, and more particularly to a system, method, and computer program product for automated reconciliation between two differently recorded sets of transaction records.

2. Description of Background Art

Software for managing personal and/or business finances and investments typically provides functionality for reconciling a user-maintained transaction register with a statement provided by a bank or other financial institution. In conventional financial software applications, a reconcile function operates as follows. Upon receiving a statement from the financial institution, the user activates the reconcile function or mode, and specifies a closing date and ending balance for the statement. All transactions in the register are displayed, except those that have previously been cleared against prior statements. Transactions occurring after the closing date may be omitted or displayed differently from other transactions. The user is then given an

opportunity to clear individual transactions in the register by matching them against transactions in the statement. Once the user has cleared all the transactions from the statement, software 307 compares the statement ending balance, adjusted for uncleared transactions, against the register ending balance. If necessary, and if requested by the user, an adjustment is made to the register. The reconciliation is then complete.

The above-described process entails substantial manual effort by the user. The user must manually review each transaction in the statement and check it against the list of register transactions. This process may be time-consuming and error-prone.

Accordingly, many financial software applications provide an automated reconciliation function, wherein the statement from the financial institution is retrieved (such as from an online source) and downloaded onto the user's computer. Software 307 then automatically compares the retrieved statement with the user-entered register. Software 307 attempts to match downloaded statement transactions with register transactions, by comparing the dates and amounts of the transactions, as well as payee or other descriptive information. Some software applications perform a "fuzzy" match determination, in order to allow for slight differences in dates, amounts, and other descriptive information. For example, such software applications may be able to detect a match between a $36.50 register transaction dated June 24th and a $36.60 statement transaction dated June 26th (thus accounting for bank processing time, inaccuracies in entry

of data, and the like). In general, such software applications attempt to mimic the type of judgment call a human would make in performing manual reconciliation. In such software applications, the user may be queried as to whether to match a particular statement transaction against one of a limited group of register transactions (or vice versa), particularly when the "fuzzy" match determination algorithm fails to provide a definitive result.

In general, such prior art systems attempt to take into account discrepancies between user-entered transactions and corresponding bank-entered transactions. However, there are many circumstances in which these systems are not capable of identifying matches, despite the "fuzzy" match capability. In particular, in many situations the bank may combine several transactions that the user has entered separately. Conversely, the user-entered register may contain a combined transaction record reflecting several transactions that the bank has entered separately. Since in such situations the amounts for the transactions differ substantially between the user-entered register and the bank statement, the automated reconciliation function is not able to detect a match.

For example, a user may enter into the software two transactions on a particular date, representing a $457 deposit and a $213 deposit. In the bank's records, these transactions may be recorded as a single transaction in the amount of $670; this may occur, for example, if the user presented the items together when making the deposit at the bank. The above-described automated

reconciliation function would fail to detect a match, since no single transaction in the register matches a transaction in the statement. This is undesirable, since the user would then have to manually indicate that these transactions should be cleared.

As another example, in an investment account, a register may record transactions representing user contributions, and may further record separate transactions representing an employer's matching contributions. The financial institution may record the user contribution and the employer contribution together, as a single transaction. Again, the above-described automated reconciliation function would fail to detect a match, since the financial institution's single transaction would not match any single transaction in the register.

What is needed, then, is a system, method, and computer program product for detecting matches between two lists of transactions, where one of the transaction lists contains a combined transaction representing two or more transactions in the other transaction list. What is further needed is an automated reconciliation system, method, and computer program product that is capable of recognizing matches between a combined transaction and separately recorded transactions. What is further needed is an automated reconciliation system, method, and computer program product that is capable of recognizing matches between different combinations of transactions.

SUMMARY OF THE INVENTION

The present invention reconciles transactions in transaction lists, and is able to detect combined transactions in one transaction list and match them appropriately with separately recorded transactions in another transaction list. The invention thus facilitates automated reconciliation between a user-maintained register and a downloaded statement from a financial institution, even when the register and the statement each combine transactions in different ways. The present invention is applicable to any software application or system having an automated reconciliation function, including for example personal finance software for tracking bank accounts and/or investment accounts.

The present invention further detects a match between a value in one list and a combination of values in a second list. The invention is thus applicable to other domains, systems, and environments in addition to automated reconciliation applications.

The invention determines whether, for two lists of values, a single value in the first list matches a combination (such as a sum) of values in the second list. In addition, the invention is able to perform many-to-many transaction matching, in which a combination of values in the first list matches a different combination of values in the second list.

In one embodiment, the invention employs a recursive function, which calls itself in order to successively eliminate individual values from the second list available for matching, until a matching combination is found, or until all

possibilities are exhausted. Values may correspond, for example, to transaction amounts for a reconciliation application.

The present invention may be implemented together with a "fuzzy" transaction-matching scheme that detects matching transactions even when particulars of the transactions are not identical, such as for example transactions having slightly different dates or amounts.

BRIEF DESCRIPTION OF THE DRAWINGS

Fig. 1 is a flowchart of a method for matching transactions according to the present invention.

Fig. 2 is a flowchart of a recursive function.

Figs. 3A and 3B are block diagrams of one system architecture for practicing the present invention.

Fig. 4 is an example of a reconciliation operation.

DETAILED DESCRIPTION OF THE PREFERRED EMBODIMENTS

Referring now to Figs. 3A and 3B, there are shown block diagrams depicting a system architecture for practicing the present invention.

In one embodiment, user's computer 304 runs software application 307 for entering and storing information describing various transactions. Examples of such a software application 307 include a personal finance software

application, an accounting system, or an investment tracking application. Software application 307 is stored in memory (not shown), such as random-access memory (RAM), and executed by a processor (not shown) such as an Intel Pentium processor. Transaction information is stored on user's computer 304 in a transaction register 305, which may be stored, for example, on a hard drive or other medium. Transaction register 305 typically contains a number of transaction records that have been entered by the user, such as for example records of checks that have been written, deposits made, and the like.

In an alternative embodiment (not shown), software 307 for entering and storing the transactions may be provided at a remote server, such as bank server 301, with which the user's computer 304 communicates over a network such as the Internet. In such a server-based software application implementation, transaction register 305 may be stored at the remote server rather than on user's computer 304. The user may interact with the server-based software application using a browser or other means for communicating over the network. Alternatively, user-entered transactions may be temporarily stored on user's computer 304 and periodically uploaded to the remote server. Such server-based software and storage may be advantageous, for example, when the user may wish to access transaction register 305 from different computers at different times.

Periodically, the user may wish to reconcile transaction register 305, containing user-entered transactions, with a statement or listing of stored

transactions 302. Such reconciliation is useful, for example, to determine whether a deposit was properly credited, or a check cashed by its payee. In an automated reconciliation scheme, user's computer 304 contacts a bank server 301 and downloads a subset of stored transactions 302 (constituting a downloadable bank statement) from server 301. Bank server 301 may be a remote computer, for example, containing records of transactions associated with many different customers. In alternative environments, server 301 may be associated with an investment firm or other financial institution. Stored transactions 302 represent those particular transactions associated with the user performing the reconciliation. The mechanisms and protocols for downloading transaction data are well known in the art (such as, for example, the Open Financial Exchange (OFX) protocol).

Software 307 then compares downloaded transactions 306 against user-entered transactions in register 305. Matching transactions are so indicated. Matches are detected by comparing, for example, amounts, reference numbers, dates, payee information, or any combination thereof. "Fuzzy" matching, taking into account slight discrepancies between the two sources of transaction information, may be employed. In some situations, register 305 may contain two or more transactions which, when combined, correspond to a single transaction in downloaded transactions 306. In other situations, the converse may be true, so that downloaded transactions 306 may contain two or more transactions which, when combined, correspond to a single transaction in

register 305. As described in more detail below, the present invention detects such matches where transactions that may appear separately in one list are combined in the other.

One skilled in the art will recognize that the system architecture illustrated in Figs. 3A and 3B is merely exemplary, and that the invention may be practiced and implemented using many other architectures and environments. For example, the present invention may be employed to match values between any two lists of transactions or other records, whether stored in the same location, entered by a user, or transmitted across a network.

Referring now to Fig. 4, there is shown an example of a reconciliation operation according to the present invention. Downloaded transactions 306 are reconciled against user-entered transaction register 305. If desired, transactions 306 may be limited to a particular subset of all recorded transactions, based on a date range. The particular fields shown in Fig. 4 for each recorded transaction include date, reference number, payee/description, and amount; other fields may be provided, such as category, comments, and the like. As can be seen from the example of Fig. 4, some transactions are reconciled by matching a combination of the stored fields. For example, transactions 306 include a Nov. 1 payment to Safeway, having a reference number of 2131 and an amount of $112.23. A transaction in register 305 has the same date, payee, reference number, and amount; therefore software 307 considers this transaction to be a match, and automatically reconciles it. Other methods and techniques for

matching single transactions may also be applied, as will be apparent to one skilled in the art.

In some cases, some of the fields may have different information, or may be missing information entirely. Software 307 employs "fuzzy" matching by allowing for certain tolerances in the values stored in the various fields of the transaction records. For example, the Nov. 11 transaction having reference number 2135 and an amount of $25.00 (as recorded in transactions 306) is reconciled with a transaction in register 305 having a slightly different date of Nov. 10 and a slightly different amount of $24.00. Since the reference numbers correspond with one another, and since the difference in date and amount are relatively slight, the match is likely correct. The degree to which such variations are tolerated, and the interaction between different fields, may be predefined or user-adjustable. In particular, dates recorded in the user-entered transaction register 305 may not necessarily correspond with dates recorded in transactions 306, due to bank processing time; thus software 307 allows for a variation in the dates of recorded transactions. One skilled in the art will recognize that other similar techniques may also be employed.

In some cases, a downloaded transaction in 306 does not correspond to a single transaction in register 305, but does correspond to a combination of transactions in register 305. This may occur, for example, if the user enters two or more transactions separately, while the bank or financial institution's records show a single combined transaction representing the same activity as the two or

more transactions entered by the user. For example, in Fig. 4 the three deposits indicated in register 305 as occurring on Nov. 1 were recorded separately by the user, but were combined into a single transaction (dated Nov. 3) in the bank's records 306. Accordingly, when software 307 fails to detect a match for a particular transaction in transactions 306, it searches for a combination of transactions in register 305 that matches the transaction in 306, in accordance with the methods of the present invention.

Conversely, a single transaction in register 305 may not correspond to a single transaction in 306, but may correspond to a combination of transactions in 306. This may occur, for example, if the user enters a combined transaction while the bank or financial institution's records show two or more distinct transactions representing the same activity as the single user-entered transaction. For example, in Fig. 4 the two checks dated Nov. 7 and Nov. 9 in transactions 305 were recorded separately by the bank, but were combined into a single transaction (dated Nov. 6) in the user-entered register 305. Accordingly, in one embodiment, the present invention may also search for a combination of transactions in 306 that matches a single transaction in register 305.

Referring now to Fig. 1, there is shown a flowchart of a method of matching such transactions according to one embodiment of the present invention. In one embodiment, the steps of Fig. 1 are performed by user's computer 304 or by a server computer, in accordance with stored instructions of software application 307. The actual location of execution of the steps of Fig. 1 is

immaterial to the invention. For example, the present invention can be implemented as an automatic reconciliation feature in a personal finance software application, on an investment website, in an accounting application, and the like. In the following discussion, descriptions of the software or function performing particular steps of the invention are intended to refer to steps performed by some computer in accordance with the stored software instructions, or alternatively to refer to any automated scheme or technique for performing a series of steps.

The user initiates a reconciliation mode or feature. Software 307 then downloads or otherwise obtains a bank statement containing a list of transactions. Alternatively, if software 307 is running on bank server 301 the bank statement may already be available to software 307, in which case the user-maintained register is instead uploaded from user's computer 304. The list of transactions from the bank statement may be a subset of all transactions, based on date range, category, and the like. The list is reconciled with a user-maintained register, as follows.

Software 307 matches 102 individual transactions between the downloaded list and a second list derived from the user-maintained register. As described above, this step is performed by, for each transaction in one list, searching for a transaction having identical or similar recorded information in the other list. Once individual transactions have been matched, software 307 determines 103 whether any unmatched transactions remain. If no unmatched

transactions remain, matches are output 109 (or marked as reconciled, as appropriate).

If software 307 determines 103 that unmatched transactions remain, it selects 104 an unmatched transaction in one of the transaction lists. The selected transaction is designated as "x" for purposes of the flowchart of Fig. 1, and the transaction list from which the transaction is selected is designated the "first list." (Thus, either the downloaded transaction list or the user-maintained register may be the first list.) The value of transaction x is stored 105 in variable v, and a counter n (representing a date range for determining matches) is initialized at zero. These designations and variable names are arbitrary, and are provided herein for illustrative purposes only.

In one embodiment, software 307 locates additional transactions occurring on the same day (or within a predefined time period) as transaction x, and adds 115 the values of any such transactions to v. Thus, v represents the total value of a number of transactions occurring on the specified day or within the predefined time period of the date of transaction x. In another embodiment, however, each such transaction is handled separately. Step 115 is not necessary to practice the present invention.

Software 307 then generates 106 a list L of unmatched transactions remaining in the second list, that occur within n days of transaction x. List L thus represents a set of candidate transactions that may, in some combination, match transaction x. As n is incremented in later steps (described below), the

date range expands so that a larger list L of candidate transactions may be considered. However, the step of successively incrementing n to consider larger date ranges is not necessary to the present invention, and the invention may be practiced with only a single fixed date range, or with no date range at all.

Once value v and list L have been determined, software 307 calls 107 a recursive function (designated "CreateSumList"), using v and L as input to the function. The recursive function, whose operation is described in more detail below, searches list L for a set of transactions having values that, in combination, match value v. If a NULL result is received 108, no match was found. Software 307 then increments 112 the value of n, and determines 113 whether n exceeds a predefined maximum date range. If n does not exceed the date range, steps 106 through 108 are repeated with the new value of n. If n does exceed the date range, no match is found 114 for transaction x, and software 307 returns to step 103 to determine whether any unmatched transactions remain.

In one embodiment, if no unmatched transactions remain in the first list, but unmatched transactions remain in the second list, steps 104 through 108 may be repeated, reversing the designations of first and second list. In other words, if the designation of "first list" and "second list" are first applied to the user-entered register and the downloaded statement, respectively, this designation is reversed when the steps are performed for the second time. In

this manner, transaction combinations in both lists can be detected and reconciled.

If in step 108, a result other than NULL is received, the received result is recorded 111 as a match. This received result indicates a set of transactions in the second list that, in combination, match transaction x; i.e., the transactions have values that add up to equal value v. In one embodiment, the set of transactions may equal or approximate value v, in accordance with a "fuzzy" matching scheme. Accordingly, software 307 may indicate that transaction x is reconciled, or matched, with the set of transactions received as a result of the recursive function.

Referring now to Fig. 2, there is shown a flowchart of a recursive function as called in step 107 of the above-described method. The function takes as input a value v of a transaction, and a transaction list L. If no match is found, the function returns a NULL value. If a match is found, the function returns a list containing either a single transaction that matches value v, or two or more transactions that, in combination, match value v.

In one embodiment, the function determines 202 whether list L is too large to be processed, and if so, returns 203 NULL.

The function then determines 204 whether any individual item in list L matches value v, and if so, returns 205 a list containing the matching item. In one embodiment, a "fuzzy" matching method may be employed in connection with step 204, so as to take into account slight differences in values or other

transaction information, as is known in the art. In another embodiment, the "fuzzy" matching method is employed only if no matching list is found after the steps of Fig. 2 have been performed.

If, in 204, no item in list L matches value v, the function determines 206 whether list L contains only one item. If so, this item must not match v, and NULL is returned 207.

If, in 206, list L contains more than one item, the function sets 208 an index i to zero. The function then subtracts 209 the value of the i^{th} entry of list L from v, and assigns this value to v'. It makes a copy 210 of list L, omitting the i^{th} entry, and designates this list as L'.

The function then recursively calls itself 211, using v' and L' as input. In this way, the function removes one transaction from consideration, and searches for a match between the remaining transactions and a modified value that takes into account the removal of the transaction. If a non-NULL result is received 212, a match has been found between v' and some combination of transactions in L'. The i^{th} entry, that was previously omitted from L', is added 213 to the list received from the recursive call. The list, including the added i^{th} entry, is then returned 214.

If a NULL result is received in 212, the function increments 215. If more entries exist 216 in list L, steps 209 through 216 are repeated. If no more entries exist 216, a NULL result is returned 217.

Using a recursive function such as that described in connection with Fig. 2, the present invention successively traverses various combinations of transactions in order to find a match for reconciliation.

In an alternative embodiment, if two or more transactions in one of the lists remain unmatched, the above-described technique is applied to a combination of two or more remaining unmatched transactions. In this way, the invention can detect a match of two or more combined transactions in one list with a different combination of transactions in the other list. For example, a deposit of $20 combined with a deposit of $10 could be matched with two deposits of $15, since each combination yields a total deposit of $30.

From the above description, it will be apparent that the invention disclosed herein provides a novel and advantageous system and method for automated reconciliation between two sets of transaction records. The foregoing discussion discloses and describes merely exemplary methods and embodiments of the present invention. As will be understood by those familiar with the art, the invention may be embodied in other specific forms without departing from the spirit or essential characteristics thereof. For example, the present invention may be practiced with specific steps and techniques that differ from those depicted in Figs. 1 and 2, such as for example employing a non-recursive function. Alternatively, other architectures and environments, including both networked and non-networked implementations, may be provided. Accordingly, the disclosure of the present invention is intended to be

illustrative, but not limiting, of the scope of the invention, which is set forth in the following claims.